

明細書

キャッシュメモリ及びその制御方法

技術分野

[0001] 本発明は、プロセッサのメモリアクセスを高速化するためのキャッシュメモリおよびその制御方法に関する。

背景技術

[0002] 近年のマイクロプロセッサでは、例えば、SRAM(Static Random Access Memory)等から成る小容量で高速なキャッシュメモリをマイクロプロセッサの内部、もしくはその近傍に配置し、データの一部をキャッシュメモリに記憶することによって、マイクロプロセッサのメモリアクセスを高速化させている。

[0003] コンピュータシステムでは、中央処理装置からキャッシュメモリへのリードアクセスまたはライトアクセスがミスヒットした場合に、主記憶装置から新たに読み出されたデータの一部が、エントリ(登録項目)としてキャッシュメモリの空きブロックに格納される。この時、空きブロックが存在しない場合には、複数のブロックのいずれか1つを選択し、選択されたブロックに格納されているエントリを主記憶装置に戻してブロック内を空き状態にし、この空きブロックに新たに読み出したデータを格納するエントリ置換処理が必要になる。上記エントリ置換処理では、最も以前に参照したデータを格納しているブロックを選択する手法、すなわち、LRU(Least Recently Used)デコード方式が一般的に採用されている。このLRUデコード方式によってキャッシュメモリの使用効率が向上し、その結果、マイクロプロセッサの実行速度が向上する。

[0004] マイクロプロセッサが処理するプログラムの中には、アクセス頻度は少ないものの、ひとたび起動された場合には高速に処理しなければならないような特殊な処理と、アクセス頻度は多いが、実行速度がそれほど要求されないような処理とが存在する。

[0005] そこで、これに対応するために例えば特許文献1等の従来技術では、キャッシュメモリにフリーズ機能を設けている。フリーズ機能は、アクセスは少ないので、ひとたび起動された場合には高速に処理しなければならないようなプログラムを予めキャッシュメモリ内にコピーしておき、その領域を書き換え禁止にしておく機能である。この

機能を有することで、コンピュータシステムは、必要な時にプログラムをキャッシュメモリから読み出して実行することができ、これにより実行時間が短縮する。また、ページ機能は、アクセス頻度が多いが、実行速度がそれほど要求されないようなプログラムやデータをキャッシュメモリ内に保存しておくことなく、その領域を解放する機能である。この機能を有することで、キャッシュメモリに余裕ができ、優先度の高い他のプログラムやデータをキャッシュメモリに取り込むことができ、これにより、キャッシュメモリの利用効率が向上し、総合的な実行時間が短縮する。

特許文献1:特開2000-200221号公報

発明の開示

発明が解決しようとする課題

- [0006] しかしながら、フリーズ機能を使用してもページ機能を使用しても、キャッシュミス発生によりリプレースする場合に無駄なリプレースをしたり、無駄なライトバックをする場合があるという問題がある。
- [0007] 例えば、無駄なリプレースが生じる場合として、プロセッサが配列要素を全てライトする場合がある。この場合、メモリから新たにデータをキャッシュメモリにリプレースしても、プロセッサから全てライトされるので、リプレースが無駄に終わる。
- [0008] また、無駄なライトバックが生じる場合として、キャッシュエントリーが単にワークデータとして使用される場合がある。この場合、最終的に破棄してもよいデータであるにも拘らず、無駄なライトバックが発生する。

課題を解決するための手段

- [0009] 本発明の目的は、無駄なリプレースや無駄なライトバックを防止するキャッシュメモリを提供することにある。
- [0010] 上記課題を解決するため本発明のキャッシュメモリは、キャッシュの単位データを保持するキャッシュエントリーに対応させて、当該キャッシュエントリーが有効であるか否かを示すバリッドフラグと、当該キャッシュエントリーに対する書き込みがなされたか否かを示すダーティフラグとを保持するフラグ保持手段と、プロセッサから指示に基づき、前記バリッドフラグおよびダーティフラグの少なくとも一方をキャッシュエントリーの状態に反して改変する改変手段とを備える。

- [0011] 前記改変手段は、キャッシングエントリーに対してタグとしてのアドレスを設定し、有効なデータを保持していない状態でバリッドフラグをセットする構成としてもよい。
- [0012] この構成によれば、配列などのデータを書き込むためのキャッシングエントリーをキャッシングメモリ上に確保することができ、しかも、上書きされる運命にあるデータをメモリからキャッシングメモリへ無駄にロードすることを防止することができる。
- [0013] また、前記改変手段は、プロセッサからの指定されたキャッシングエントリーに対してライトバックされていない書き換えられたデータを保持している状態でダーティフラグをリセットする。
- [0014] この構成によれば、テンポラリーなワークデータ等破棄される運命にあるデータを保持するキャッシングエントリーから無駄にライトバックすることを防止することができる。
- [0015] ここで、前記キャッシングメモリは、さらに、プロセッサから指定されたアドレス範囲を保持する保持手段と、保持されたアドレス範囲に属するデータを保持するキャッシングエントリーを特定する特定手段とを備え、前記改変手段は、特定されたキャッシングエントリーに対して前記バリッドフラグおよびダーティフラグの少なくとも一方を改変する構成としてもよい。
- [0016] この構成によれば、プロセッサから指定されたアドレス範囲で、キャッシングメモリへの無駄なデータのロード又はメモリへの無駄なライトバックを防止することができる。
- [0017] ここで、前記特定手段は、前記アドレス範囲の先頭アドレスがラインデータの途中を指す場合、当該先頭アドレスを、前記アドレス範囲に含まれる先頭のラインを指すスタートラインアドレスに変換する第1変換手段と、前記アドレス範囲の末尾アドレスがラインデータの途中を指す場合、当該末尾アドレスを、前記アドレス範囲に含まれる末尾のラインを指すエンドラインアドレスに変換する第2変換手段と、前記スタートラインアドレスからエンドラインアドレスまでの各ラインアドレスに対応するデータを保持するキャッシングエントリーがあるか否かを判定する判定手段と備える構成としてもよい。
- [0018] この構成によれば、プロセッサは、前記アドレス範囲として、キャッシングメモリのラインサイズ及びライン境界のアドレスとは無関係に任意のアドレスから任意のアドレス(又は任意のサイズ)を指定することができる。つまり、プロセッサにおいてキャッシングメモリのラインサイズ及びライン境界のアドレスを管理する必要がないので、キャッシングメモリ

管理のための負荷を解消することができる。

- [0019] ここで、前記改変手段は、ダーティフラグのリセット指示付きメモリアクセス命令が実行されたことを検出する命令検出手段と、当該命令によってアクセスがなされたキャッシュエントリーに対してダーティフラグをリセットするフラグ書き換え手段とを備える構成としてもよい。
- [0020] ここで、前記改変手段は、バリッドフラグのリセット指示付きメモリアクセス命令が実行されたことを検出する命令検出手段と、当該命令によってアクセスがなされたキャッシュエントリーに対してバリッドフラグをリセットするフラグ書き換え手段とを備える構成としてもよい。

また、本発明のキャッシュメモリの制御方法についても上記と同様の手段、作用を有する。

- [0021] 本発明のキャッシュメモリによれば、配列などのデータを書き込むためのキャッシュエントリーをキャッシュメモリ上に確保することができ、しかも、上書きされる運命にあるデータをメモリからキャッシュメモリへ無駄にロードすることを防止することができる。

また、テンポラリーなワークデータ等破棄される運命にあるデータを保持するキャッシュエントリーから無駄にライトバックすることを防止することができる。
- [0022] さらに、プロセッサは、前記アドレス範囲として、キャッシュメモリのラインサイズ及びライン境界のアドレスとは無関係に任意のアドレスから任意のアドレス(又は任意のサイズ)を指定することができる。つまり、プロセッサにおいてキャッシュメモリのラインサイズ及びライン境界のアドレスを管理する必要がないので、キャッシュメモリ管理のための負荷を解消することができる。

発明の効果

- [0023] 本発明のキャッシュメモリによれば、プロセッサの指示に基づいて、配列などのデータを書き込むためのキャッシュエントリーをキャッシュメモリ上に確保することができ、しかも、上書きされる運命にあるデータをメモリからキャッシュメモリへ無駄にロードすることを防止することができる。また、テンポラリーなワークデータ等破棄される運命にあるデータを保持するキャッシュエントリーから無駄にライトバックすることを防止することができる。

図面の簡単な説明

[0024] [図1]本発明の実施の形態1におけるプロセッサ、キャッシングメモリ、メモリを含むシステムの概略構成を示すブロック図である。

[図2]キャッシングメモリの構成例を示すブロック図である。

[図3]キャッシングエントリーの詳細なビット構成を示す図である。

[図4]制御部の構成を示すブロック図である。

[図5]フラグ改変部の構成例を示すブロック図である。

[図6](a)スタートアドレスレジスタにスタートアドレスを書き込む命令の一例を示す。(b)サイズレジスタにサイズを書き込む命令の一例を示す。(c)コマンドレジスタにコマンドを書き込む命令の一例を示す。(d)コマンドの一例を示す。

[図7]スタートアライナ及びエンドアライナの説明図である。

[図8]フラグ書換部407におけるフラグ改変処理の一例を示すフローチャートである。

[図9]本発明の実施の形態2におけるキャッシングメモリの構成を示すブロック図である。

[図10]キャッシングエントリーのビット構成を示す。

[図11]制御部の構成を示すブロック図である。

[図12]フラグ更新部による使用フラグUの更新例を示す。

[図13](a)ウイークフラグが存在しない場合にキャッシングエントリーがリプレースされる様子を示す図である。(b)リプレース処理におけるウイークフラグWの役割を示す説明図である。

[図14]リプレース部におけるUフラグ更新処理を示すフローチャートである。

[図15]リプレース部におけるリプレース処理を示すフローチャートである。

[図16]フラグ改変部の構成を示すブロック図である。

[図17]コマンドフォーマットの一例を示す。

[図18]フラグ書換部におけるWフラグ設定処理の一例を示すフローチャートである。

符号の説明

[0025]

1	プロセッサ
2	メモリ
3	キャッシングメモリ

- 20 アドレスレジスタ
- 21 メモリI/F
- 30 デコーダ
- 31a~31d ウェイ
- 32a~32d 比較器
- 33a~33d アンド回路
- 34 オア回路
- 35 セレクタ
- 36 セレクタ
- 37 デマルチプレクサ
- 38 制御部
- 39 フラグ更新部
- 40 リプレース部
- 41 フラグ改変部
- 131a~131d ウェイ
- 138 制御部
- 139 フラグ更新部
- 140 リプレース部
- 141 フラグ改変部
- 401 コマンドレジスタ
- 402 スタートアドレスレジスタ
- 403 サイズレジスタ
- 404 加算器
- 405a スタートアライナ
- 405b スタートアライナ
- 406a エンドアライナ
- 406b エンドアライナ
- 407 フラグ書換部

407a フラグ書換部

発明を実施するための最良の形態

[0026] (実施の形態1)

<全体構成>

図1は、本発明の実施の形態1におけるプロセッサ1、キャッシングメモリ3、メモリ2を含むシステムの概略構成を示すブロック図である。同図のように、本発明のキャッシングメモリ3は、プロセッサ1およびメモリ2を有するシステムに備えられる。

キャッシングメモリ3は、キャッシングエントリー毎にキャッシングエントリーが有効であるか否かを示すバリッドフラグVと、当該キャッシングエントリーに対して書き込みがなされたか否かを示すダーティフラグDとを有し、プロセッサ1により指定されたアドレスのデータを保持するキャッシングエントリーに対してデータをライトバックすることなくダーティフラグDを1から0(ダーティでない)に改変し、また、プロセッサ1により指定されたアドレスに対応するキャッシングエントリーを確保してメモリからデータをロードすることなくバリッドフラグVを1(有効)に改変するよう構成されている。

ダーティフラグDを1から0に改変するのは、最終的には破棄されるテンポラリーナンクデータを保持するキャッシュエントリーから無駄にライトバックすることを防止するためである。また、キャッシングエントリーを確保してメモリからデータをロードすることなくバリッドフラグVを1(有効)に改変することにより、配列など書き込みをするためのキャッシングエントリーを予め確保するためである。

[0027] <キャッシングメモリの構成>

以下、キャッシングメモリ3の具体例として、4ウェイ・セット・アソシエイティブ方式のキャッシングメモリに本発明を適用した場合の構成について説明する。

図2は、キャッシングメモリ3の構成例を示すブロック図である。同図のように、キャッシングメモリ3は、アドレスレジスタ20、メモリI/F21、デコーダ30、4つのウェイ31a～31d(以下ウェイ0～3と略す)、4つの比較器32a～32d、4つのアンド回路33a～33d、オア回路34、セレクタ35、36、デマルチプレクサ37、制御部38を備える。

[0028] アドレスレジスタ20は、メモリ2へのアクセスアドレスを保持するレジスタである。このアクセスアドレスは32ビットであるものとする。同図に示すように、アクセスアドレスは、

最上位ビットから順に、21ビットのタグアドレス、4ビットのセットインデックス(図中のSI)、5ビットのワードインデックス(図中のWI)を含む。ここで、タグアドレスはウェイにマッピングされるメモリ中の領域(そのサイズはセット数×ブロックである)を指す。この領域のサイズは、タグアドレスよりも下位のアドレスビット(A10～A0)で定まるサイズつまり2kバイトであり、1つのウェイのサイズでもある。セットインデックス(SI)はウェイ0～3に跨る複数セットの1つを指す。このセット数は、セットインデックスが4ビットなので16セットある。タグアドレスおよびセットインデックスで特定されるキャッシュエントリーは、リプレース単位であり、キャッシュメモリに格納されている場合はラインデータ又はラインと呼ばれる。ラインデータのサイズは、セットインデックスよりも下位のアドレスビットで定まるサイズつまり128バイトである。1ワードを4バイトとすると、1ラインデータは32ワードである。ワードインデックス(WI)は、ラインデータを構成する複数ワード中の1ワードを指す。アドレスレジスタ20中の最下位2ビット(A1、A0)は、ワードアクセス時には無視される。

- [0029] メモリI/F21は、キャッシュメモリ3からメモリ2へのデータのライトバックや、メモリ2からキャッシュメモリ3へのデータのロード等、キャッシュメモリ3からメモリ2をアクセスするためのI/Fである。
- [0030] デコーダ30は、セットインデックスの4ビットをデコードし、4つのウェイ0～3に跨る16セット中の1つを選択する。
- [0031] 4つのウェイ0～3は、同じ構成を有数する4つのウェイであり、 $4 \times 2k$ バイトの容量を有する。各ウェイは、16個のキャッシュエントリーを有する。
- [0032] 図3に1つのキャッシュエントリーにおける詳細なビット構成を示す。同図のように、1つのキャッシュエントリーは、バリッドフラグV0～V3、21ビットのタグ、128バイトのラインデータ、ダーティフラグD0～D3を有する。
- [0033] タグは21ビットのタグアドレスのコピーである。
ラインデータは、タグアドレスおよびセットインデックスにより特定されるブロック中の128バイトデータのコピーであり、32バイトの4つのサブラインからなる。
- [0034] バリッドフラグV0～V3は、4つのサブラインに対応し、サブラインが有効か否かを示す。

[0035] ダーティフラグD0～D3は、4つのサブラインに対応し、そのサブラインにプロセッサから書き込みがあったか否か、つまりサブライン中にキャッシュされたデータが存在するが書き込みによりメモリ中のデータと異なるためメモリに書き戻すことが必要か否かを示す。

[0036] 比較器32aは、アドレスレジスタ20中のタグアドレスと、セットインデックスにより選択されたセットに含まれる4つのタグ中のウェイ0のタグとが一致するか否かを比較する。比較器32b～32cについても、ウェイ31b～31dに対応すること以外は同様である。

[0037] アンド回路33aは、バリッドフラグと比較器32aの比較結果とが一致するか否かを比較する。この比較結果をh0とする。比較結果h0が1である場合は、アドレスレジスタ20中のタグアドレスおよびセットインデックスに対応するラインデータが存在すること、つまりウェイ0においてヒットしたことを意味する。比較結果h0が0である場合は、ミスヒットしたことを意味する。アンド回路33b～33dについても、ウェイ31b～31dに対応すること以外は同様である。その比較結果h1～h3は、ウェイ1～3でヒットしたかミスしたかを意味する。

[0038] オア回路34は、比較結果h0～h3のオアをとる。このオアの結果をhitとする。hitは、キャッシュメモリにヒットしたか否かを示す。

[0039] セレクタ35は、選択されたセットにおけるウェイ0～3のラインデータのうち、ヒットしたウェイのラインデータを選択する。

[0040] セレクタ36は、セレクタ35により選択された32ワードのラインデータにうち、ワードインデックスに示される1ワードを選択する。

[0041] デマルチプレクサ37は、キャッシュエントリにデータを書き込む際に、ウェイ0～3の1つに書き込みデータを出力する。この書き込みデータはワード単位でよい。

[0042] 制御部38は、キャッシュメモリ3の全体の制御を行う。特に、プロセッサからのコマンド及びアドレス指定に従って、Vフラグの改変と、Dフラグの改変とを行う。

[0043] <制御部の構成>

図4は、制御部38の構成を示すブロック図である。同図のように、制御部38は、フラグ更新部39、リプレース部40、フラグ改変部41とを含む。

フラグ更新部39は、従来技術と同様にキャッシュメモリの状態を反映するようVフラ

グ、Dフラグの更新を行う。

リプレース部40は、従来技術と同様にキャッシングエントリーのリプレースを行う。

フラグ改変部41は、プロセッサ1からのコマンド及びアドレス指定に応じて、Vフラグの改変とDフラグの改変とを行う。このコマンドには、Vフラグ設定コマンドとDフラグ設定コマンドとがある。

[0044] <フラグ改変部の構成>

図5は、フラグ改変部41の構成例を示すブロック図である。同図のようにフラグ改変部41は、コマンドレジスタ401、スタートアドレスレジスタ402、サイズレジスタ403、加算器404、スタートアライナ405a、405b、エンドアライナ406a、406b、フラグ書換部407を備える。

- [0045] コマンドレジスタ401は、プロセッサ1から直接アクセス可能なレジスタであり、プロセッサ1により書き込まれたコマンドを保持する。図6(c)に、コマンドレジスタ401にコマンドを書き込む命令の一例を示す。この命令は、通常の転送命令(mov命令)であり、ソースオペランドとしてコマンドを、デスティネーションオペランドとしてコマンドレジスタ(CR)401を指定している。図6(d)に、コマンドフォーマットの一例を示す。このコマンドフォーマットは、Vフラグ設定コマンド用の2ビットのフィールドと、Dフラグ設定コマンド用の2ビットのフィールドと、サブライン指定用の4ビットのフィールドとを含む。
- [0046] 例えば、Vフラグ設定コマンドは、「10」であればV=0に設定することを指示し(Vリセットコマンド)、「11」であればV=1に設定することを指示し(Vセットコマンド)、「00」であれば無効(ノーオペレーション)を意味する。Dフラグ設定コマンドも同様である。サブライン指定フィールドは、サブラインを個別に指定するためのフィールドであり、個別指定がない場合はライン(全サブライン)指定であるものとする。
- [0047] スタートアドレスレジスタ402は、プロセッサ1から直接アクセス可能なレジスタであり、プロセッサ1により書き込まれたスタートアドレスを保持する。このスタートアドレスはCフラグを設定すべきアドレス範囲の開始位置を示す。図6(a)に、スタートアドレスレジスタ402にスタートアドレスを書き込む命令の一例を示す。この命令も、図6(c)と同様に通常の転送命令(mov命令)である。
- [0048] サイズレジスタ403は、プロセッサ1から直接アクセス可能なレジスタであり、プロセッ

サ1により書き込まれたサイズを保持する。このサイズは、スタートアドレスからのアドレス範囲を示す。図6(b)に、サイズレジスタ403にサイズを書き込む命令の一例を示す。この命令も、図6(c)と同様に通常の転送命令(mov命令)である。なお、サイズの単位は、バイト数であっても、ライン数(キャッシュエントリー数)であってもよく、予め定められた単位であればよい。

- [0049] 加算器404は、スタートアドレスレジスタ402に保持されたスタートアドレスとサイズレジスタ403に保持されたサイズとを加算する。加算結果は、アドレス範囲の終了位置を指すエンドアドレスである。加算器404は、サイズがバイト数指定の場合はバイトアドレスとして加算し、サイズがライン数指定の場合はラインアドレスとして加算すればよい。
- [0050] スタートアライナ405a、405bは、スタートアドレスをライン境界の位置に調整する。スタートアライナ405aはエンドアドレスの方向に、405bはエンドアドレスとは反対の方向に調整する。この調整によりプロセッサ1はラインサイズ及びライン境界とは無関係に任意のアドレスをスタートアドレスとして指定することができる。
- [0051] エンドアライナ406a、406bは、エンドアドレスをライン境界の位置に調整する。エンドアライナ4065aはスタートアドレスの方向に、406bはスタートアドレスとは反対の方向に調整する。この調整によりプロセッサ1はラインサイズ及びライン境界とは無関係に任意の大きさを上記サイズとして指定することができる。
- [0052] 図7に、スタートアライナ405a、405b及びエンドアライナ406a、406bの説明図を示す。同図において、プロセッサ1から指定されたスタートアドレスはラインNの途中の任意の位置を指す。スタートアライナ405aは、次のライン(N+1)の先頭を指すよう調整し、調整後のアドレスをアラインスタートアドレスaとして出力する。スタートアライナ405bは、スタートアドレスのデータを含むラインNの先頭を指すよう調整し、調整後のアドレスをアラインスタートアドレスbとして出力する。アラインスタートアドレスが指すラインをスタートラインと呼ぶ。
- [0053] また、エンドアドレスはラインMの途中の任意の位置を指す。エンドアライナ406aは、直前のライン(M-1)の先頭を指すよう調整し、調整後のアドレスをアラインエンドアドレスaとして出力する。エンドアライナ406bは、エンドアドレスのデータを含むライ

ンMの先頭を指すよう調整し、調整後のアドレスをアラインエンドアドレスbとして出力する。アラインエンドアドレスが指すラインをエンドラインと呼ぶ。

- [0054] 同図のように、スタートアライナ405a及びエンドアライナ406aはライン単位で内側アラインを行う。スタートアライナ405b及びエンドアライナ406bはライン単位で外側アラインを行う。さらに、ライン単位の外側アラインの後、さらに、サブライン単位の外側アラインと内側アラインが可能である。
- [0055] フラグ書換部407は、スタートラインからエンドラインまで、コマンドに従ってVフラグ又はDフラグの値を設定する。その際、スタートライン及びエンドラインが内側アラインか外側アラインかはコマンドに応じて選択される。
- [0056] <フラグ改変処理>

図8は、フラグ書換部407におけるフラグ改変処理の一例を示すフローチャートである。同図では、DリセットコマンドとVセットコマンドとを示している。

同図において、コマンドレジスタ401にDリセットコマンドが書き込まれている場合(S80)、フラグ書換部407は、スタートアライナ405a、405b、エンドアライナ406a、406bの出力の中から内側アラインによるスタートラインとエンドラインとを選択する(S81)。ここで内側アラインを選択しているのは、ラインNとラインMにおけるスタートアドレスとエンドアドレスの外側のデータは、プロセッサ1により破棄されるとは限らないからである。

- [0057] さらに、フラグ書換部407は、スタートラインからエンドラインまでの各ラインアドレスを順に出力しながらループ1の処理(S82～S86)を行う。フラグ書換部407は、各ラインについて同じ処理を行うので、ここでは1ライン分の処理について説明する。
- [0058] すなわち、フラグ書換部407は、キャッシュメモリ3がプロセッサ1からアクセスされていない間に、ラインアドレスをアドレスレジスタ20に出力し(S83)、アドレスレジスタ20のタグアドレスとキャッシュエントリーのタグとを比較器32a～32dに比較させ、ヒットするかどうかを判定する(S84)。さらにフラグ書換部407は、ヒットした場合には、ヒットしたキャッシュエントリーに対してD0～D3フラグを0にリセットし(S85)、ミスヒットした場合には、キャッシュメモリにエントリーされていないのでなにもしない。
- [0059] このように、スタートラインからエンドラインまでの各ラインについて、キャッシュメモリ

3にエントリーされている場合には、D0～D3フラグに0が設定される。フラグ書換部407は、ループ1の終了後、コマンドレジスタ401のコマンドをクリアする(S98)。これにより、テンポラリーなワークデータ等破棄される運命にあるデータを保持するキャッシュエントリーから無駄にライトバックすることを防止することができる。

- [0060] また、図8において、コマンドレジスタ401にVセットコマンドが書き込まれている場合(S87)、フラグ書換部407は、スタートアライナ405a、405b、エンドアライナ406a、406bの出力の中から外側アラインによるスタートラインとエンドラインとを選択する(S88)。ここで外側アラインを選択しているのは、内側アラインではスタートアドレスからエンドアドレスまでのサイズを確保できないからである。
- [0061] さらに、フラグ書換部407は、スタートラインからエンドラインまでの各ラインアドレスを順に出力しながらループ2の処理(S89～S97)を行う。フラグ書換部407は、各ラインについて同じ処理を行うので、ここでは1ライン分の処理について説明する。
- [0062] すなわち、フラグ書換部407は、キャッシュメモリ3がプロセッサ1からアクセスされていない間に、ラインアドレスをアドレスレジスタ20に出力し(S90)、アドレスレジスタ20のタグアドレスとキャッシュエントリーのタグとを比較器32a～32dに比較させ、ヒットするかどうかを判定する(S91)。さらにフラグ書換部407は、ヒットしなかった場合には、当該ラインアドレスに対応するセット内の4つのウェイからLRU方式でリプレース対象のウェイを1つ選択し(S92)、選択されたウェイのダーティフラグD0～D3の論理和が1であるか否かを判定する(S93)。この論理和が1、つまりダーティであると判定された場合、フラグ書換部407は、ダーティなサブラインのみをライトバックする(S94)。上記の論理和が1でないつまりダーティでないと判定された場合、又はダーティなサブラインのライトバックの後に、フラグ書換部407は、キャッシュエントリーにメモリからデータをロードすることなく、キャッシュエントリーにラインアドレスをタグとして設定し(S95)、バリッドフラグV0～V3を1にセットする(S96)。このようにして、有効なデータを保持していないがV0～V3=1と設定された1ライン分のキャッシュエントリーが確保される。
- [0063] さらに、フラグ書換部407は、ループ1の終了後、コマンドレジスタ401のコマンドをクリアする(S98)。

[0064] これにより、配列などのデータを書き込み用のキャッシングエントリーをキャッシングメモリ上に確保し、かつ、無駄なライトバックを防止することができる。

[0065] 以上説明してきたように、本実施の形態におけるキャッシングメモリによれば、Dリセットコマンドにより、破棄される運命にあるデータを保持するキャッシングエントリーから無駄にライトバックすることを防止することができる。また、Vセットコマンドにより、配列などのデータを書き込み用のキャッシングエントリーをキャッシングメモリ上に確保し、かつ、無駄なライトバックを防止することができる。

[0066] なお、上記実施の形態では、DリセットコマンドとVセットコマンドについて説明したが、Vリセットコマンドについては、図8に示したS87～S98において、ステップS95を削除し、ステップS96においてV0～V3を0にリセットすることにより実現することができる。これによれば、無駄なライトバックを防止してキャッシングエントリーを開放することができる。

[0067] また、Dセットコマンドは、図8のS80～S86と同様に実行可能ではあるが、プロセッサ1からストア命令によりデータを書き込めばDフラグがセットされること、Dフラグのセットによりライトバック動作が発生することを考えれば、あまり実用的とはいえない。しかし、キャッシングメモリのテスト動作や性能の評価や検証等に利用することができる。

[0068] <変形例>

なお、本発明のキャッシングメモリは、上記の実施の形態の構成に限るものではなく、種々の変形が可能である。以下、変形例のいくつかについて説明する。

(1) 上記実施の形態では、V0～V3フラグ、D0～D3フラグを同時にセット又はリセットしているが、サブ単位にセット又はリセットするようにしてもよい。

例えば、スタートラインとエンドラインのみをサブライン単位で処理する場合、フラグ書換部407は、外側アライン(ライン)されたスタートアドレスとエンドアドレスを選択し、さらに外側アライン(サブライン)または内側アライン(サブライン)によりスタートラインのサブラインアドレスと、エンドラインのサブラインアドレスとを算出し、スタートラインとエンドラインのみをサブライン単位で処理すればよい。また、プロセッサ1は、コマンド中のサブライン指定フィールドにおいてその旨を指定すればよい。

例えば、プロセッサ1がサイズレジスタ403のサイズを0に指定し、コマンド中のサブ

ライン指定フィールドにおいて特定のサブラインを指定した場合に、フラグ書換部407は、指定されたサブラインのみを対象に処理をすればよい。

(2) また、フラグ改変部41は、ダーティフラグのリセット指示付きメモリアクセス命令が実行されたこと検出する命令検出部と、当該命令によってアクセスがなされたキャッシュエントリーに対してダーティフラグをリセットするフラグ書き換え手部とを備える構成としてもよい。

さらに、命令検出部は、バリッドフラグのリセット指示付きメモリアクセス命令が実行されたこと検出し、フラグ書き換え手部は、当該命令によってアクセスがなされたキャッシュエントリーに対してバリッドフラグをリセットする構成としてもよい。

(3) 上記実施の形態では、4ウェイ・セット・アソシエイティブのキャッシュメモリを例に説明したが、ウェイ数は、いくつでもよい。また、上記実施の形態では、セット数が16である例を説明したが、セット数はいくつでもよい。

(4) 上記実施の形態では、セット・アソシエイティブのキャッシュメモリを例に説明したが、フル・アソシエイティブ方式のキャッシュメモリであってもよい。

(5) 上記実施の形態では、サブラインのサイズをラインのサイズの1/4としているが、1/2、1/8、1/16等他のサイズでもよい。その場合、各キャッシュエントリーは、サブラインと同数のバリッドフラグおよびダーティフラグをそれぞれ保持すればよい。

[0069] (実施の形態2)

実施の形態1では、Vフラグの改変とDフラグの改変とを行う構成について説明したが、本実施の形態では、これ以上使用(書き込み及び読み出し)するかしないかを示すW(ウィーク)フラグを有するキャッシュメモリにおいてWフラグを改変する構成について説明する。

[0070] <キャッシュメモリの構成>

図9は、本発明の実施の形態2におけるキャッシュメモリの構成を示すブロック図である。同図のキャッシュメモリは、図2の構成と比較して、ウェイ31a～31dの代わりにウェイ131a～131dを備える点と、制御部38の代わりに制御部138を備える点とが異なっている。以下、同じ点は説明を省略して、異なる点を中心に説明する。

[0071] ウェイ131aは、ウェイ31aと比べて、各キャッシュエントリー中にWフラグ及びUフラ

グが追加されている点が異なる。ウェイ131b～131dも同様である。

- [0072] 図10に、キャッシュエントリーのビット構成を示す。1つのキャッシュエントリーは、バリッドフラグV0～V3、21ビットのタグ、128バイトのラインデータ、ウィークフラグW、使用フラグU及びダーティフラグD0～D3を保持する。
- [0073] このうち、ウィークフラグWは、プロセッサからのアクセスに関しては、これ以上使用するか否かを意味し、キャッシュメモリにおけるリプレース制御に関しては、他のキャッシュエントリーよりも真っ先に追い出してもよい最弱のリプレース対象を意味する。このように、ウィークフラグWは二つの意味を有することから、クリーニング処理とリプレース処理との2つの処理で参照される。
- [0074] 使用フラグUは、そのキャッシュエントリーにアクセスがあったか否かを示し、LRU方式におけるミスヒットによるリプレースに際して4つのウェイのキャッシュエントリー間におけるアクセス順序データの代わりに用いられる。より正確には、使用フラグUの1は、アクセスがあったことを、0はないことを意味する。ただし、1つのセット内の4つウェイの使用フラグが全て1になれば、0にリセットされる。別言すれば、使用フラグUは、アクセスされた時期が古いか新しいか2つの相対的な状態を示す。つまり、使用フラグUが1のキャッシュエントリーは、使用フラグが0のキャッシュエントリーよりも新しくアクセスされたことを意味する。
- [0075] 制御部138は、制御部38と比べて、Wフラグを設定する点と、LRU方式におけるアクセス順序情報の代わりに使用フラグUを用いる点とが異なる。
- [0076] <制御部の構成>

図11は、制御部138の構成を示すブロック図である。同図の制御部138は、制御部38と比較して、フラグ更新部39、リプレース部40、フラグ改変部41の代わりにフラグ更新部139、リプレース部140、フラグ改変部141を備える点が異なる。

- [0077] フラグ更新部139は、フラグ更新部39と同様にVフラグ、Dフラグを更新することに加えて、キャッシュメモリがアクセスされたときに使用フラグUの更新処理を行う。
- [0078] リプレース部140は、通常のLRU方式ではなく、使用フラグUをアクセス順序とする擬似的なLRU方式によりリプレースを行う。ただし、リプレース処理に際してW=1のキャッシュエントリーは真っ先にリプレース対象として選択する。

[0079] フラグ改変部141は、プロセッサ1からのコマンドに応じてウィークフラグWを設定する。プロセッサ1は、もはや使用(書き込み及び読み出し)をしないキャッシングエントリーについてウィークフラグの設定を指示するコマンドをキャッシングメモリ3に対して発行する。W=1のキャッシングエントリーは、キャッシング時に使用フラグUの値に関わらず、真っ先にリプレース対象となる。また、W=1のキャッシングエントリーがダーティであればクリーニング処理の対象となる。

[0080] <使用フラグUの説明>

図12は、フラグ更新部39による使用フラグUの更新例を示す。同図の上段、中断、下段は、ウェイ0～3に跨るセットNを構成する4つのキャッシングエントリーを示している。4つのキャッシングエントリー右端の1又は0は、それぞれ使用フラグの値である。この4つの使用フラグUをU0～U3と記す。

[0081] 同図上段では $(U_0 \sim U_3) = (1, 0, 1, 0)$ であるので、ウェイ0、2のキャッシングエントリーはアクセスがあったことを、ウェイ1、3のキャッシングエントリーはアクセスがないことを意味する。

[0082] この状態で、メモリアクセスがセットN内のウェイ1のキャッシングエントリーにヒットした場合、同図中段に示すように、 $(U_0 \sim U_3) = (1, 1, 1, 0)$ に更新される。つまり、実線に示すようにウェイ1の使用フラグU1が0から1に更新される。

[0083] さらに、同図中段の状態で、メモリアクセスがセットN内のウェイ3のキャッシングエントリーにヒットした場合、同図下断に示すように、 $(U_0 \sim U_3) = (0, 0, 0, 1)$ に更新される。つまり、実線に示すようにウェイ3の使用フラグU1が0から1に更新される。加えて、破線に示すようにウェイ3以外の使用フラグU0～U2が1から0に更新される。これにより、ウェイ3のキャッシングエントリーが、ウェイ0～2の各キャッシングエントリーよりも新しくアクセスされたことを意味することになる。

[0084] リプレース部140は、キャッシング時にW=1のキャッシングエントリーが存在しなければ、使用フラグに基づいてリプレース対象のキャッシングエントリーを決定してリプレースを行う。例えば、フラグ更新部39は、図5上段では、ウェイ1とウェイ3の何れかをリプレース対象と決定し、図5中段ではウェイ3をリプレース対象と決定し、図5下段ではウェイ0～2の何れかをリプレース対象と決定する。

[0085] <ウィークフラグWの説明>

図13(a)は、ウィークフラグが存在しないと仮定した場合の比較例であり、キャッシュエントリーがリプレースされる様子を示す図である。同図においても、図12と同様にウェイ0～3に跨るセットNを構成する4つのキャッシュエントリーを示している。4つのキャッシュエントリー右端の1又は0は、それぞれ使用フラグの値である。また、データEのみアクセス頻度の低いデータを、データA、B、C、Dはアクセス頻度の高いデータとする。

- [0086] 同図(a)の第1段目の状態で、プロセッサ1がデータEにアクセスすると、キャッシュミスが発生する。このキャッシュミスにより、例えば、 $U=0$ のキャッシュエントリーの中からアクセス頻度の高いデータCのキャッシュエントリーがアクセス頻度の低いデータEにリプレースされ、第2段目の状態となる。
- [0087] 第2段目の状態で、プロセッサ1がデータCにアクセスすると、キャッシュミスが発生する。このキャッシュミスにより、 $U=0$ のキャッシュエントリーであるアクセス頻度の高いデータDのキャッシュエントリーがアクセス頻度の高いデータCにリプレースされ、第3段目の状態となる。
- [0088] 第3段目の状態で、プロセッサ1がデータDにアクセスすると、キャッシュミスが発生する。このキャッシュミスにより、例えば、アクセス頻度の高いデータCのキャッシュエントリーがアクセス頻度の高いデータDにリプレースされ、第3段目の状態となる。
- [0089] 同様に、第4段目でも、使用頻度の低いデータEはリプレース対象として選択されないで、キャッシュメモリーに残っている。
- [0090] 第5段目の状態で、使用頻度の低いデータEは最も古い($U=0$)ことから、リプレース対象として選択されて、追い出される。
- [0091] このように、擬似LRU方式において(通常のLRU方式においても)、アクセス頻度の低いデータEによって、4ウェイの場合には最悪4回のキャッシュミスを誘発する場合がある。

図13(b)は、リプレース処理におけるウィークフラグWの役割を示す説明図である。

- [0092] 同図(b)の第1段目の状態(同図(a)の第1段目と同じ)で、プロセッサ1がデータEにアクセスすると、キャッシュミスが発生する。このキャッシュミスにより、例えば、 $U=0$

のキャッシュエントリーの中からアクセス頻度の高いデータCのキャッシュエントリーがアクセス頻度の低いデータEにリプレースされる。このとき、プロセッサ1は、データEのキャッシュエントリーにウィークフラグWを1に設定するものとする。これにより、次のキャッシュミス時にデータEのキャッシュエントリーが真っ先に追い出され、第2段目の状態となる。

- [0093] 第2段目の状態で、プロセッサ1がデータCにアクセスすると、キャッシュミスが発生する。このキャッシュミスにより、W=1のキャッシュエントリーであるアクセス頻度の低いデータEのキャッシュエントリーがリプレース対象として選択され、アクセス頻度の高いデータCにリプレースされ、第3段目の状態となる。
- [0094] このように、ウィークフラグWを設けることにより、アクセス頻度の低いデータによるキャッシュミスの誘発を低減することができる。
- [0095] <Uフラグ更新処理>

図14は、リプレース部140におけるUフラグ更新処理を示すフローチャートである。同図では、バリッドフラグが0(無効)であるキャッシュエントリーの使用フラグUは0に初期化されているものとする。

同図において、リプレース部140は、キャッシュヒットしたとき(ステップS61)、セットインデックスにより選択されたセットにおけるヒットしたウェイの使用フラグUを1にセットし(ステップS62)、そのセット内の他のウェイの使用フラグUを読み出し(ステップS63)、読み出した使用フラグUが全て1であるか否かを判定し(ステップS64)、全て1でなければ終了し、全て1であれば他のウェイの全ての使用フラグUを0にリセットする(ステップS65)。

このようにしてリプレース部140は、図12、図13(a)(b)に示した更新例のように、使用フラグUを更新する。

- [0096] <リプレース処理>

図15は、リプレース部140におけるリプレース処理を示すフローチャートである。同図においてリプレース部140は、メモリアクセスがミスしたとき(ステップS91)、セットインデックスにより選択されたセットにおける、4つウェイの使用フラグU及びウィークフラグWを読み出し(ステップS92)、W=1のウェイが存在するか否かを判定する(ステッ

プS93)。W=1のウェイが存在しないと判定された場合、U=0のウェイを1つ選択する(ステップS94)。このとき、使用フラグUが0になっているウェイが複数存在する場合は、リプレース部140はランダムに1つを選択する。また、W=1のウェイが存在すると判定された場合、Uフラグの値に関わらずW=1のウェイを1つ選択する(ステップS95)。このとき、キークリアフラグWが1になっているウェイが複数存在する場合は、リプレース部140はランダムに1つを選択する。

[0097] さらに、リプレース部140は、当該セットにおける選択されたウェイのキャッシュエントリーを対象にリプレースし(ステップS96)、リプレース後に当該キャッシュエントリーの使用フラグUを1に、キークリアフラグWを0初期化する(ステップS97)。なお、このときバリッドフラグV、データティフラグDは、それぞれ1、0に初期化される。

このように、W=1のウェイが存在しない場合、リプレース対象は、使用フラグUが0のキャッシュエントリーの中から1つ選択される。

また、W=1のウェイが存在する場合、リプレース対象は、使用フラグUが0であると1であるとを問わず、W=1のウェイのキャッシュエントリーから1つ選択される。これにより図13(a) (b)に示したように、アクセス頻度の低いデータがキャッシュメモリに残ることによるキャッシュミスの誘発を低減することができる。

[0098] <フラグ改変部141の構成>

図16は、フラグ改変部141の構成を示すブロック図である。同図の構成は、図5に示したフラグ改変部41と比較して、フラグ書換部407の代わりにフラグ書換部407aを備える点が異なっている。

[0099] フラグ書換部407aは、フラグ書換部407の機能に加えて、Wフラグの改変と使用フラグUの改変とを行う点が異なっている。そのため、コマンドレジスタ401には、Wフラグの設定を指示するWコマンド、Uフラグの設定を指示するUコマンドがプロセッサ1によって設定可能になっている。図17に、これらのコマンドフォーマットの一例を示す。同図のコマンドフォーマットは、図6(d)に示したコマンドフォーマットに対してWコマンド及びUコマンドのフィールドが追加されている。Wコマンド及びUコマンドの内容はDコマンドやVコマンドと同様である。

[0100] <Wフラグ設定処理>

図18は、フラグ書換部407aにおけるWフラグ設定処理の一例を示すフローチャートである。

フラグ書換部407aは、コマンドレジスタ401にWフラグ設定コマンドが保持されている場合、スタートラインからエンドラインまでの各ラインアドレスを順に出力しながらループ1の処理(S82～S86)を行う。フラグ書換部407aは、各ラインについて同じ処理を行うので、ここでは1ライン分の処理について説明する。

すなわち、フラグ書換部407aは、キャッシングメモリ3がプロセッサ1からアクセスされていない間に、ラインアドレスをアドレスレジスタ20に出力し(S83)、アドレスレジスタ20のタグアドレスとキャッシングエントリーのタグとを比較器32a～32dに比較させ、ヒットするかどうかを判定する(S84)。さらにフラグ書換部407aは、ヒットした場合には、ヒットしたキャッシングエントリーに対してWフラグを1にセットし(S85)、ミスヒットした場合には、キャッシングメモリにエントリーされていないのでなにもしない。

これにより、スタートラインからエンドラインまでの各ラインについて、キャッシングメモリ3にエントリーされている場合には、Wフラグが1に設定される。

[0101] <Uフラグ設定処理>

フラグ書換部407aは、コマンドレジスタ401にUフラグ設定コマンドが保持されている場合、当該コマンドに従ってUフラグを設定する。この処理は、図18においてWフラグをUフラグと読み替えることにより、Wフラグ設定処理と全く同様に実行される。

[0102] 以上説明してきたように、本実施の形態におけるキャッシングメモリによれば、Wフラグを設定することにより、W=1のキャッシングエントリーをキャッシングミス時には最古のキャッシングエントリとして真っ先にリプレース対象とすることができる。また、使用フラグUの値は1ビットだけであるがアクセス順序が古いか新しいかをしめすので、Uフラグ設定処理によって、スタートラインからエンドラインまでの各ラインについてアクセス順序を設定することができる。例えば、プロセッサ1は、キャッシングメモリ3に残しておきたいアドレスのデータをアドレス範囲として指定してU=1を設定するUフラグ設定コマンドを発行し、

以上説明してきたように、本実施の形態におけるキャッシングメモリによれば、Wフラグを設定することにより、W=1のキャッシングエントリーをキャッシングミス時には最古のキャ

ツシュエントリとして真っ先にリプレース対象とすることができます。また、使用フラグUの値は1ビットだけであるがアクセス順序が古いか新しいかをしめすので、Uフラグ設定処理によって、スタートラインからエンドラインまでの各ラインについてアクセス順序を設定することができる。例えば、プロセッサ1は、キャッシングメモリ3に残しておきたいアドレスのデータをアドレス範囲として指定してU=1を設定するUフラグ設定コマンドを発行すればよい。逆に、キャッシングメモリ3が追い出してもよいデータをアドレス範囲として指定してU=0を設定するUフラグ設定コマンドを発行すればよい。

[0103] <変形例>

- (1) W=1のキャッシングエントリーは真っ先にリプレース対象とされるが、制御部は、リプレースされるまでの間に、ダーティであればクリーニング(ライトバック)を行ってもよい。
- (2) 図6(a)(b)(c)に示した各命令は、コンパイラによりプログラム中に挿入してもよい。その際、コンパイラは、例えば配列データの書き込みや、圧縮動画データをデコードする際のブロックデータの書き込み等、これ以上書き込みをしないプログラム位置に、上記各命令を挿入するようにすればよい。

産業上の利用可能性

[0104] 本発明は、メモリアクセスを高速化するためのキャッシングメモリに適しており、例えば、オンチップキャッシングメモリ、オフチップキャッシングメモリ、データキャッシングメモリ、命令キャッシングメモリ等に適している。

請求の範囲

[1] キャッシュの単位データを保持するキャッシュエントリーに対応させて、当該キャッシュエントリーが有効であるか否かを示すバリッドフラグと、当該キャッシュエントリーに対する書き込みがなされたか否かを示すダーティフラグとを保持するフラグ保持手段と、
プロセッサからの指示に基づき、前記バリッドフラグおよびダーティフラグの少なくとも一方をキャッシュエントリーの状態に反して改変する改変手段とを備えることを特徴とするキャッシュメモリ。

[2] 前記改変手段は、メモリからデータをロードすることなく、キャッシュエントリーに対してタグとしてのアドレスを設定しバリッドフラグをセットすることを特徴とする請求項1記載のキャッシュメモリ。

[3] 前記改変手段は、キャッシュエントリーに対してライトバックされていない書き換えられたデータを保持している状態でダーティフラグをリセットすることを特徴とする請求項2記載のキャッシュメモリ。

[4] 前記キャッシュメモリは、さらに、
プロセッサから指定されたアドレス範囲を保持する保持手段と、
保持されたアドレス範囲に属するデータを保持するキャッシュエントリーを特定する特定手段とを備え、
前記改変手段は、特定されたキャッシュエントリーに対して前記バリッドフラグおよびダーティフラグの少なくとも一方を改変することを特徴とする請求項2又は3記載のキャッシュメモリ。

[5] 前記特定手段は、
前記アドレス範囲の先頭アドレスがラインデータの途中を指す場合、当該先頭アドレスを、前記アドレス範囲に含まれる先頭のラインを指すスタートラインアドレスに変換する第1変換手段と、
前記アドレス範囲の末尾アドレスがラインデータの途中を指す場合、当該末尾アドレスを、前記アドレス範囲に含まれる末尾のラインを指すエンドラインアドレスに変換する第2変換手段と、

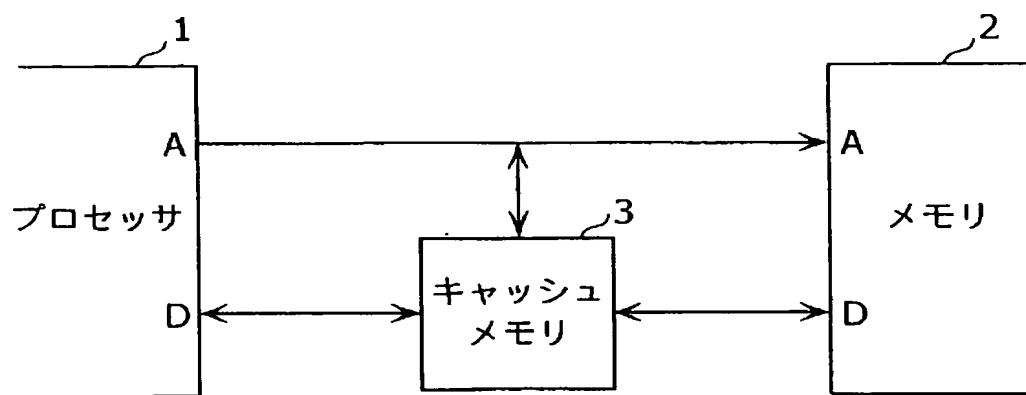
前記スタートラインアドレスからエンドラインアドレスまでの各ラインアドレスに対応するデータを保持するキャッシュエントリーがあるか否かを判定する判定手段とを備えることを特徴とする請求項4記載のキャッシュメモリ。

- [6] 前記改変手段は、
ダーティフラグのリセット指示付きメモリアクセス命令が実行されたことを検出する命令検出手段と、
当該命令によってアクセスがなされたキャッシュエントリーに対してダーティフラグをリセットするフラグ書き換え手段と
を備えることを特徴とする請求項1記載のキャッシュメモリ。
- [7] 前記改変手段は、
バリッドフラグのリセット指示付きメモリアクセス命令が実行されたことを検出する命令検出手段と、
当該命令によってアクセスがなされたキャッシュエントリーに対してバリッドフラグをリセットするフラグ書き換え手段と
を備えることを特徴とする請求項1記載のキャッシュメモリ。
- [8] キャッシュの単位データを保持するキャッシュエントリーに対応させて、当該キャッシュエントリーが有効であるか否かを示すバリッドフラグと、当該キャッシュエントリーに対する書き込みがなされたか否かを示すダーティフラグとを有するキャッシュメモリの制御方法であって、
プロセッサの指示に基づき、メモリからデータをロードすることなく、キャッシュエントリーに対してタグとしてのアドレスを設定しバリッドフラグをセットするステップと、
プロセッサの指示に基づき、キャッシュエントリーに対してライトバックされていない書き換えられたデータを保持している状態でダーティフラグをリセットするステップと
を有することを特徴とする制御方法。

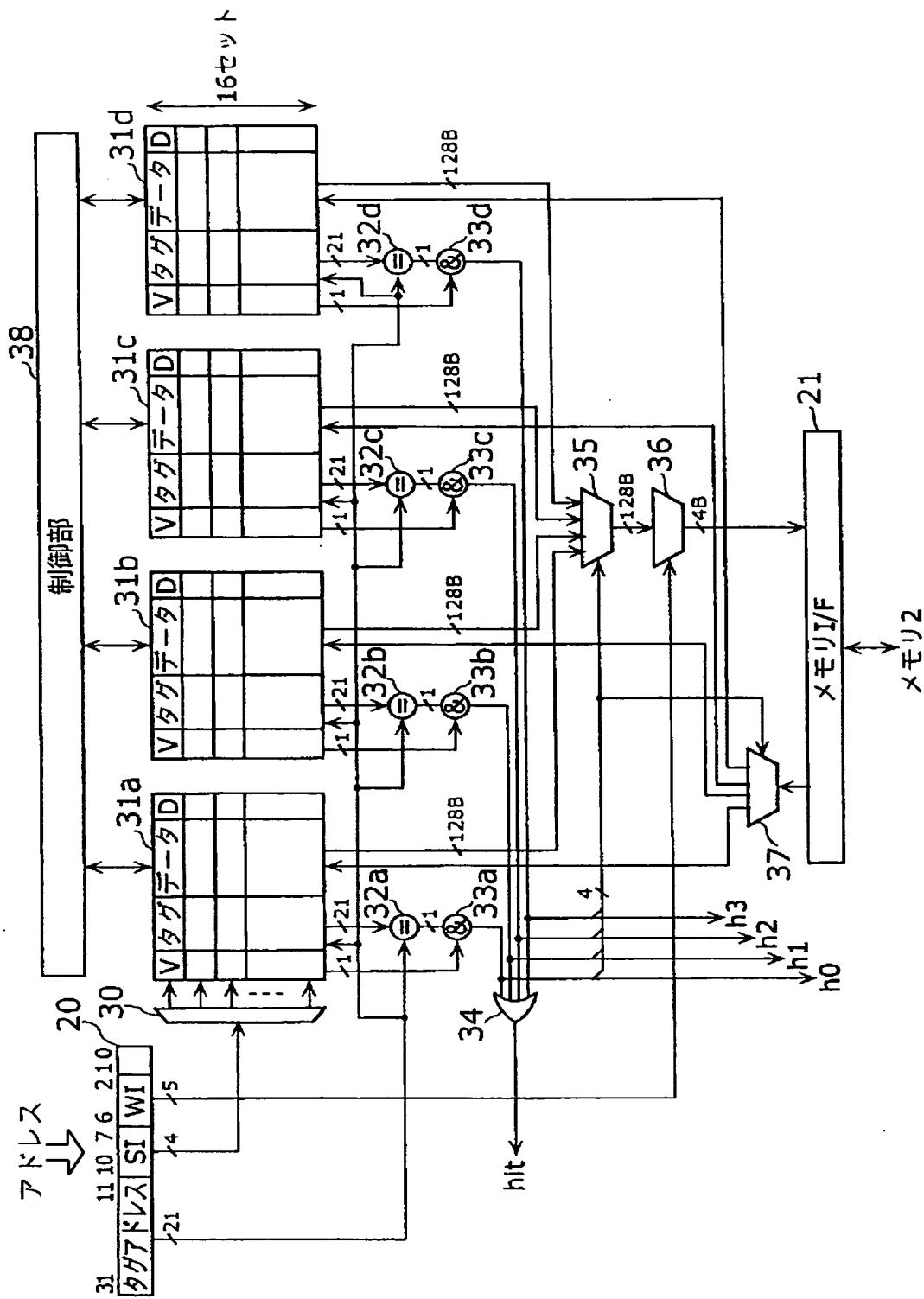
要 約 書

本発明のキャッシングメモリは、キャッシングの単位データを保持するキャッシングエントリーに対応させて、当該キャッシングエントリーが有効であるか否かを示すバリッドフラグと、当該キャッシングエントリーに対する書き込みがなされたか否かを示すダーティフラグと有するキャッシングメモリであつて、プロセッサからの指示に基づき、メモリからデータをロードすることなく、キャッシングエントリーに対してタグとしてのアドレスを設定しバリッドフラグをセットし、あるいは、キャッシングエントリーに対してライトバックされていない書き換えられたデータを保持している状態でダーティフラグをリセットするフラグ改変部を備える。

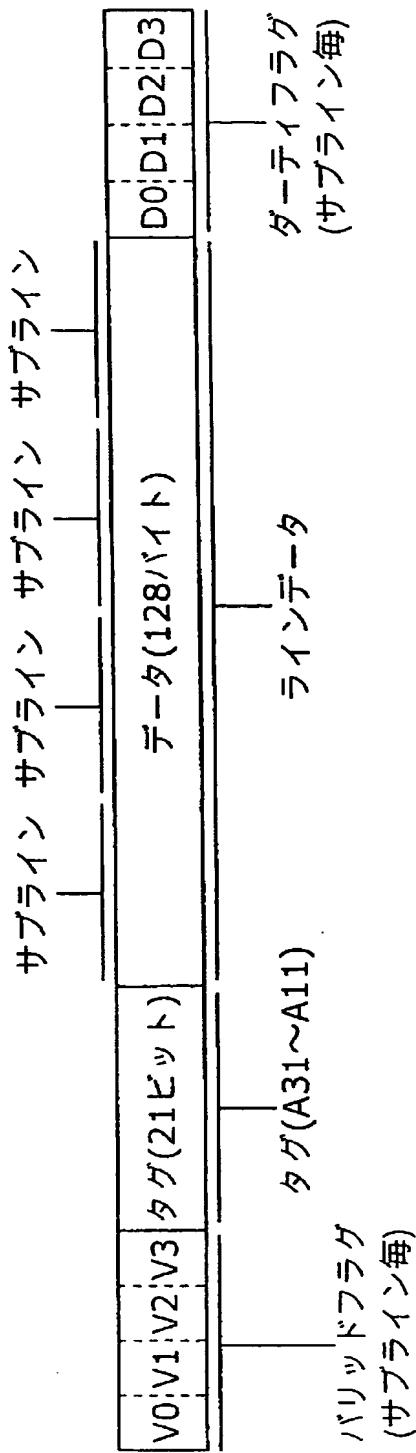
[図1]



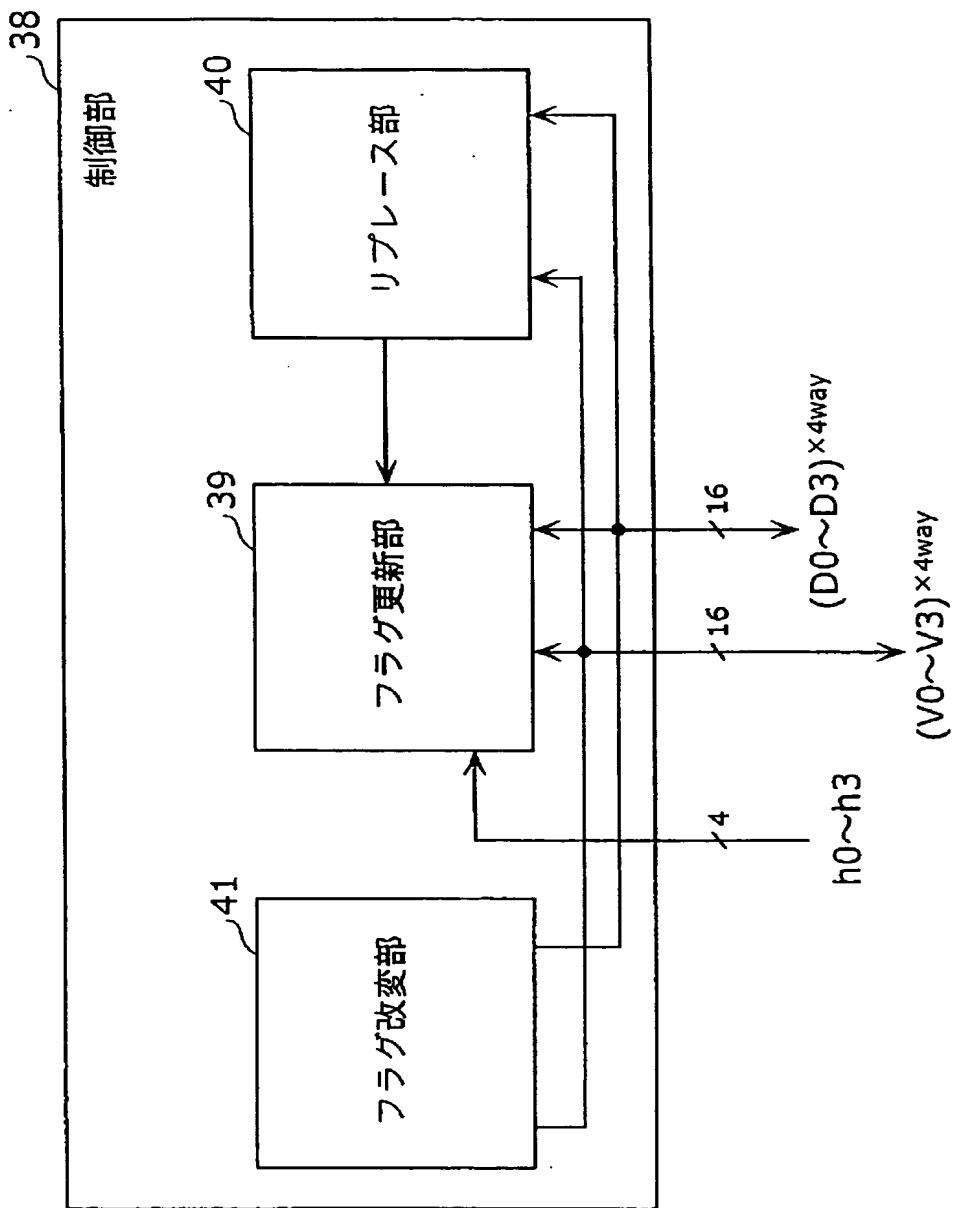
[図2]



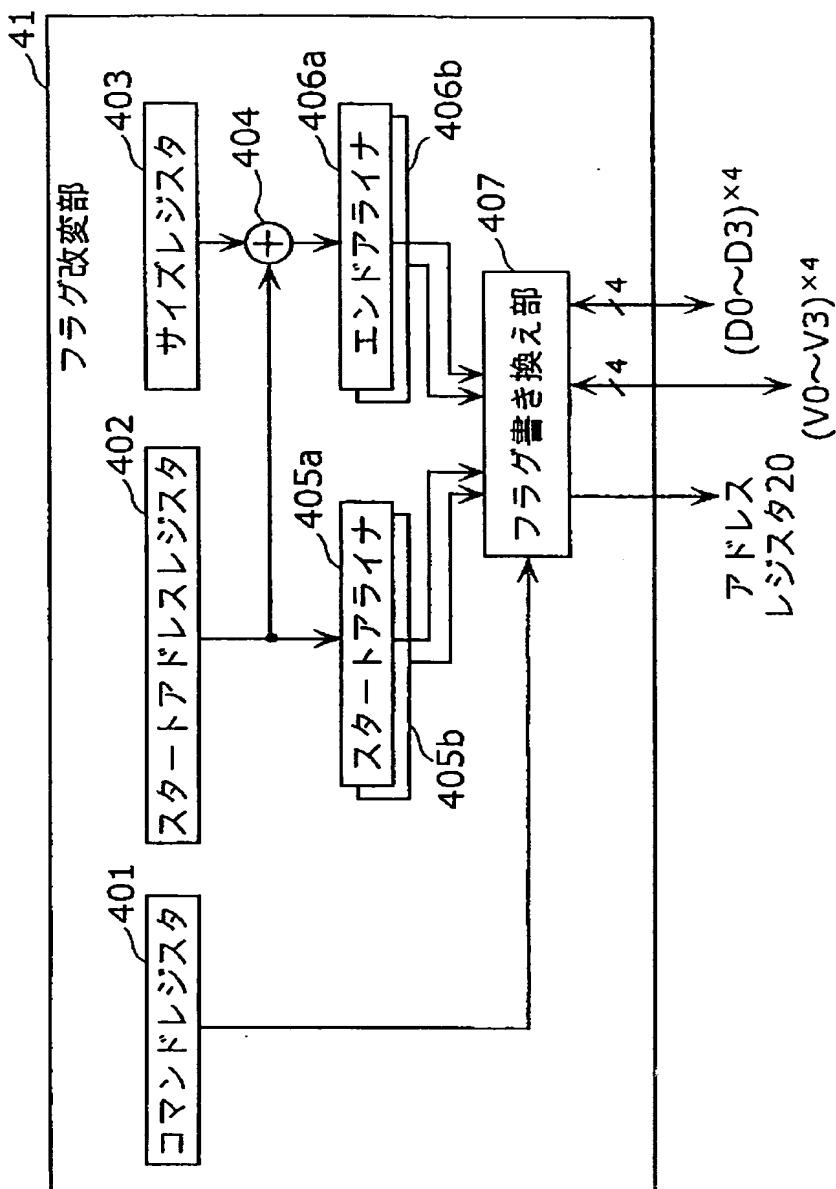
[図3]



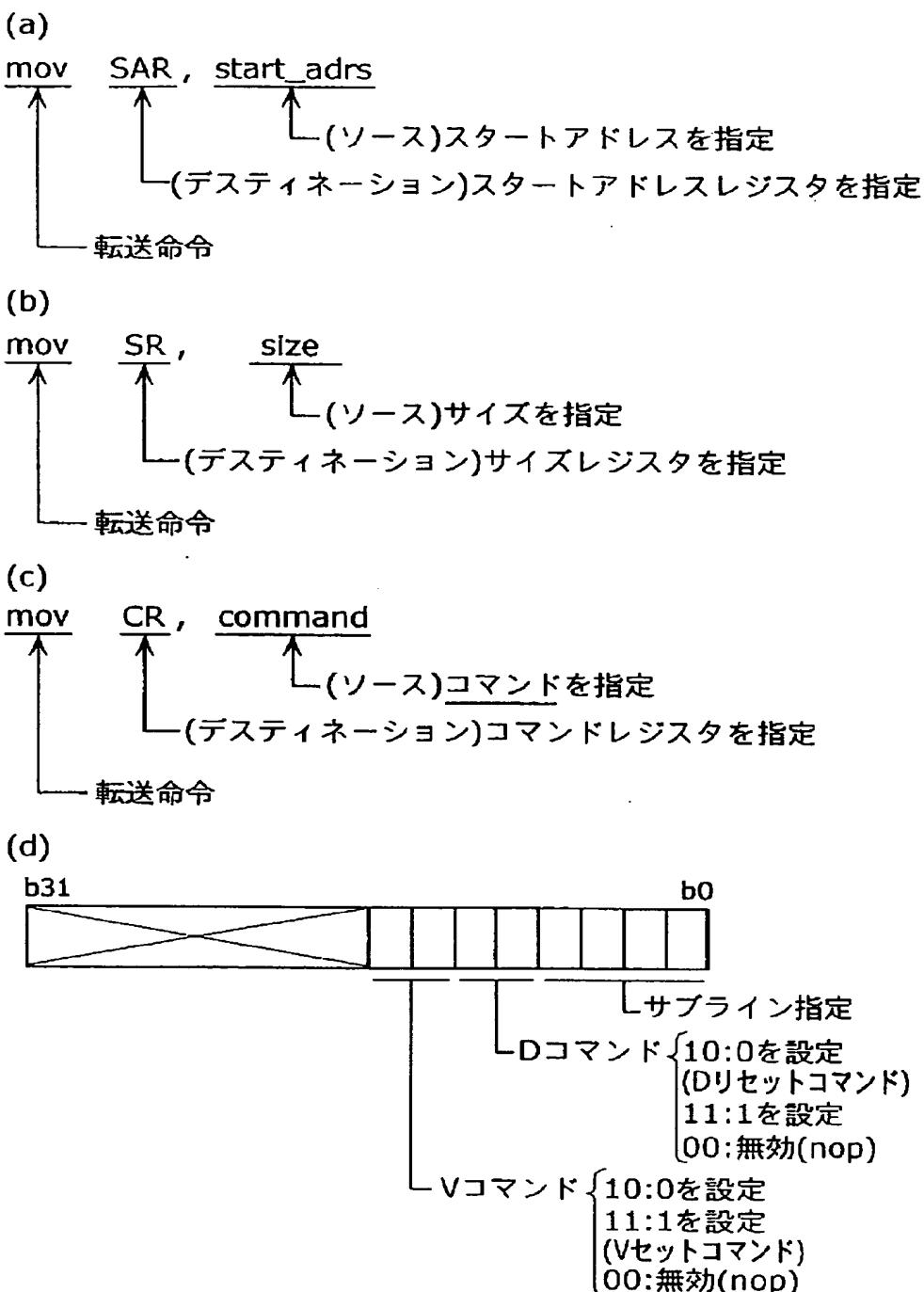
[図4]



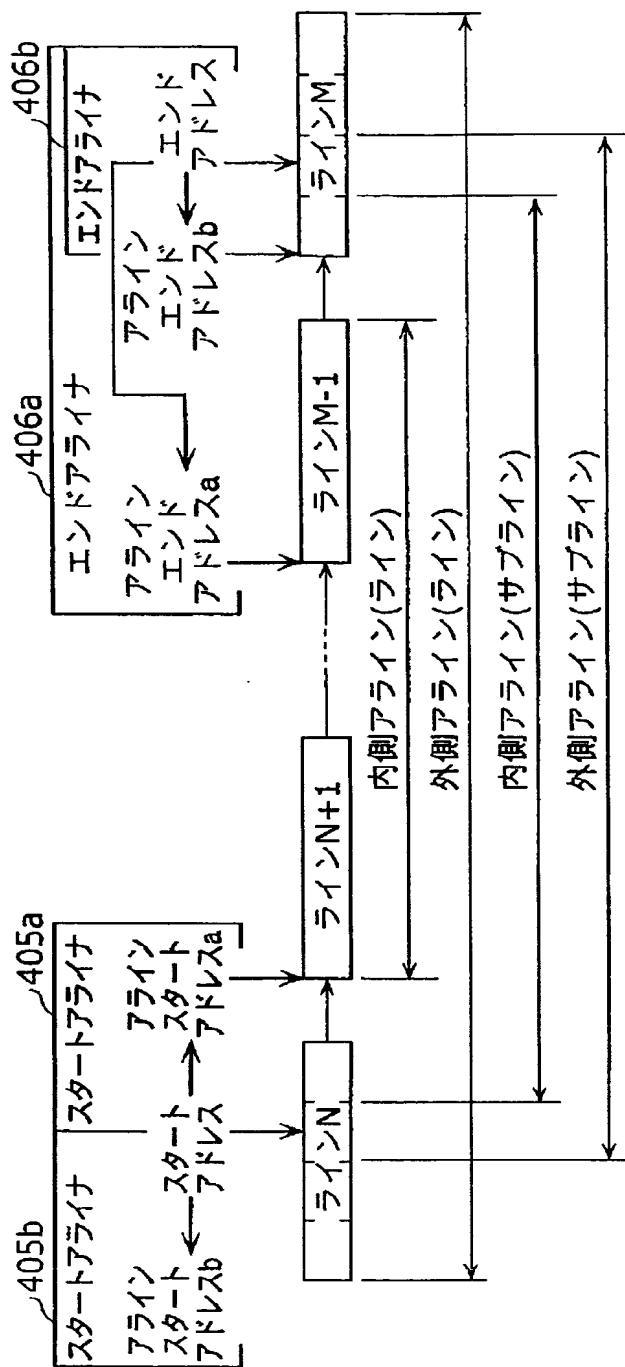
[図5]



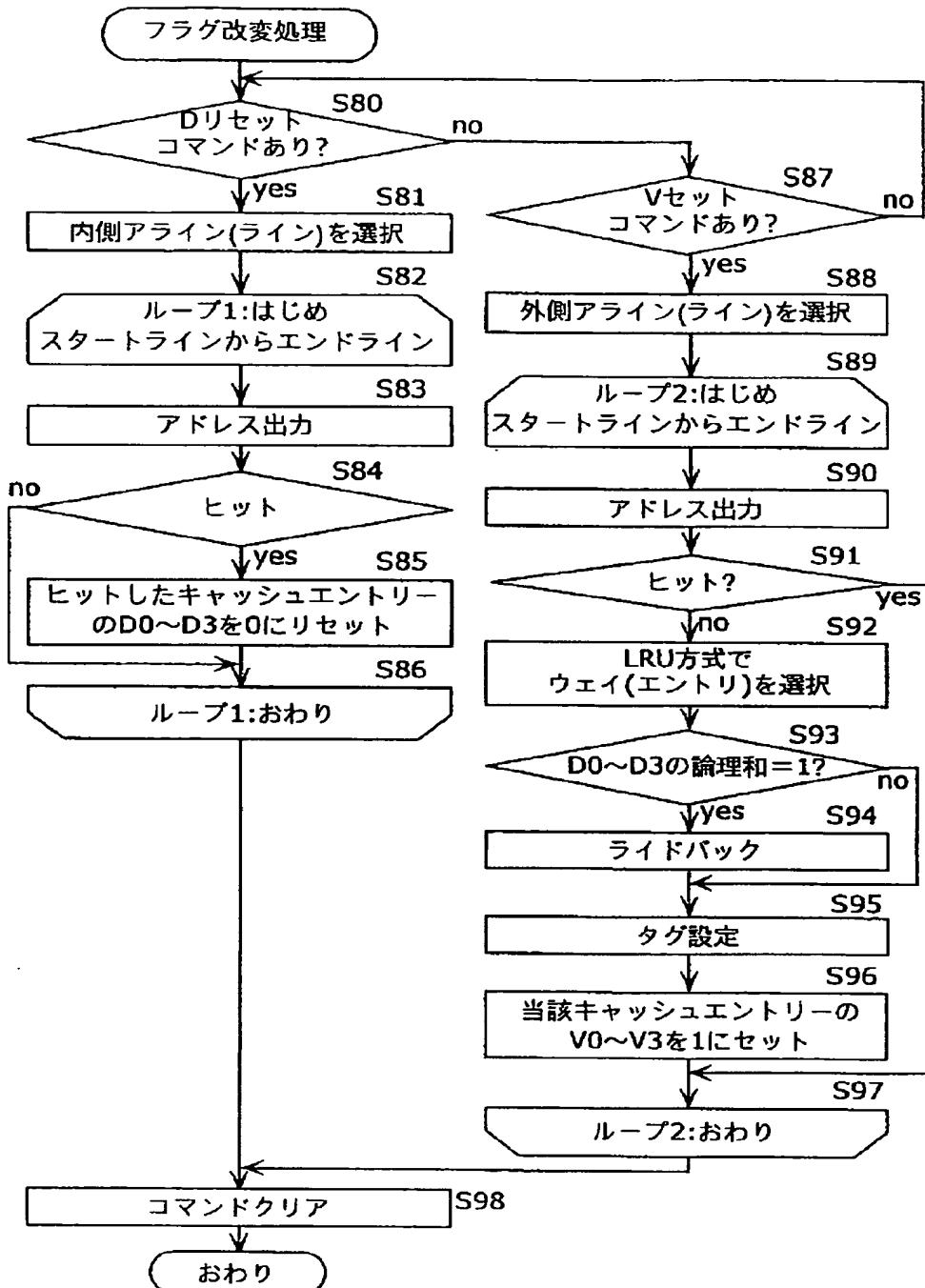
[図6]



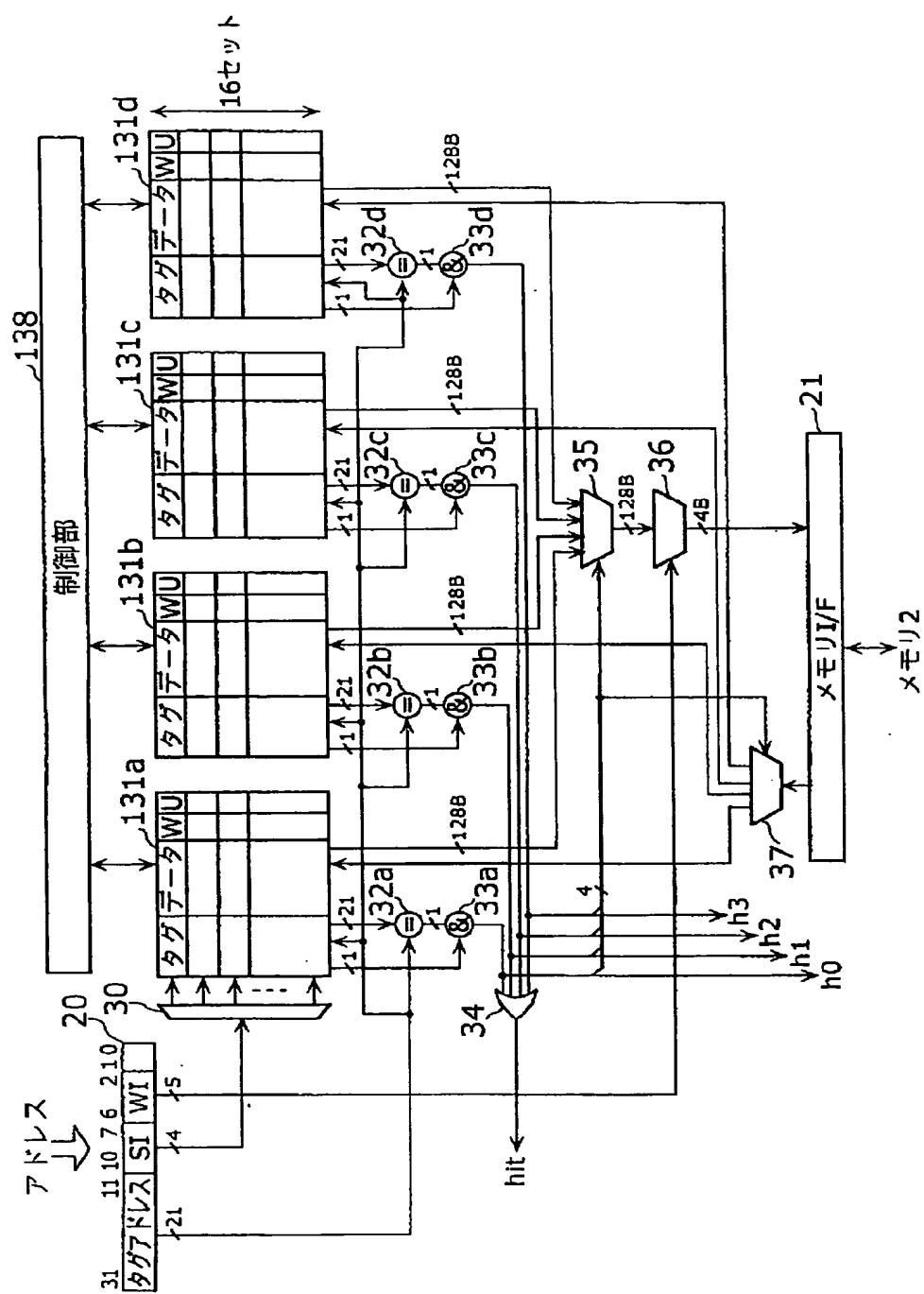
[図7]



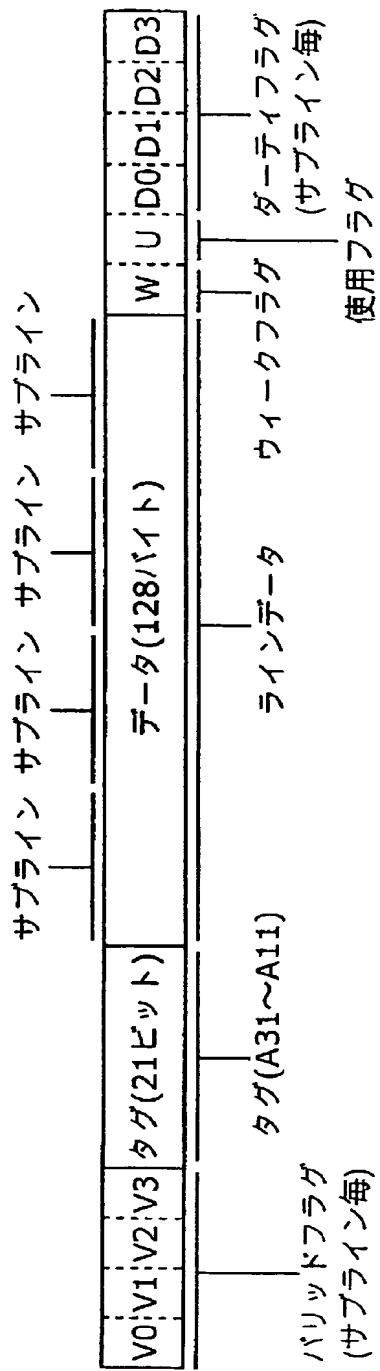
[図8]



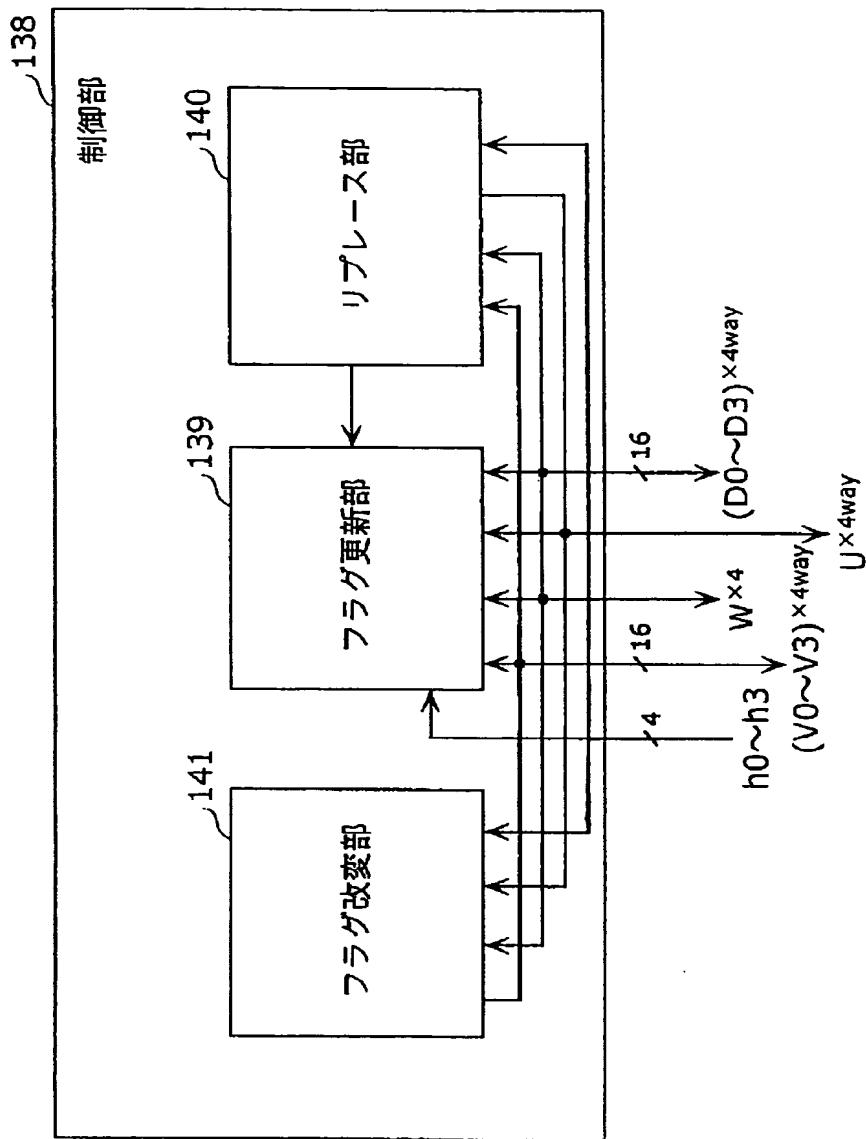
[図9]



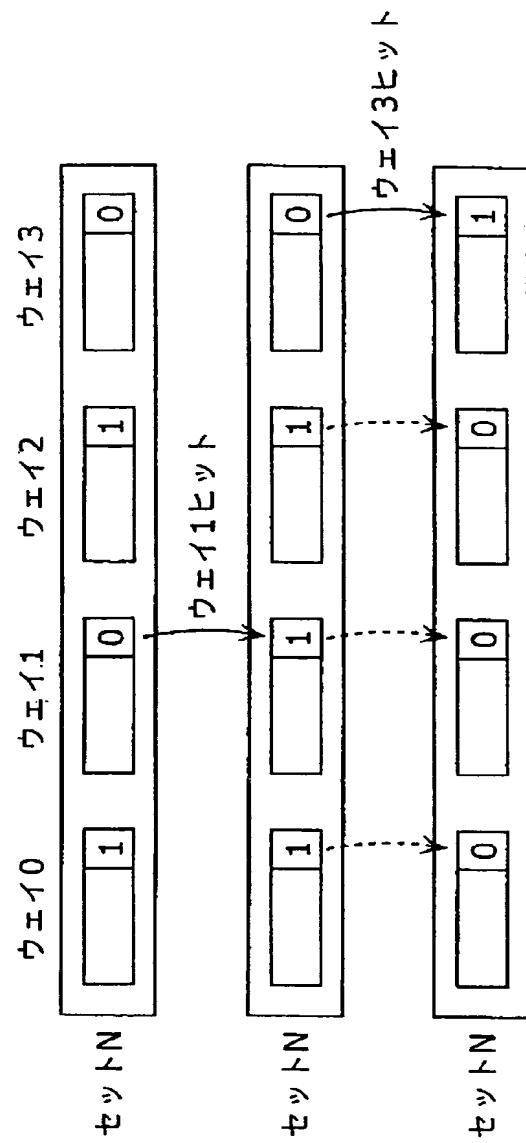
[図10]



[図11]

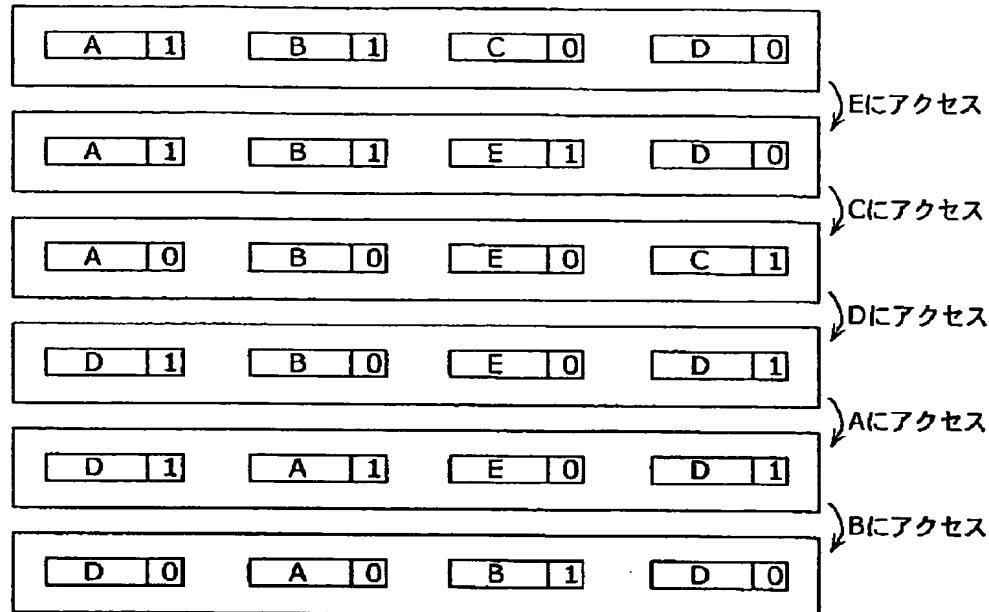


[図12]

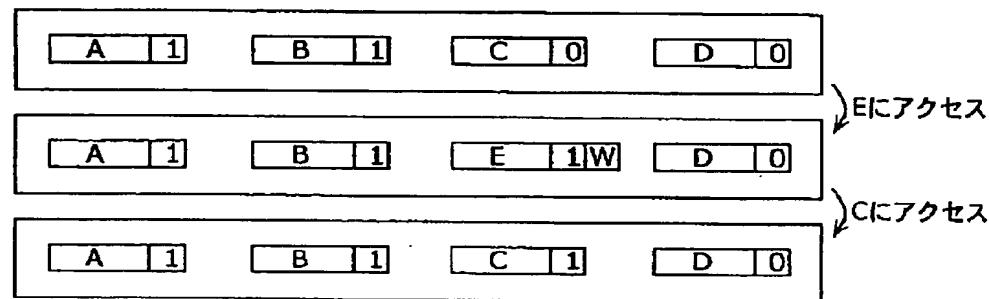


[図13]

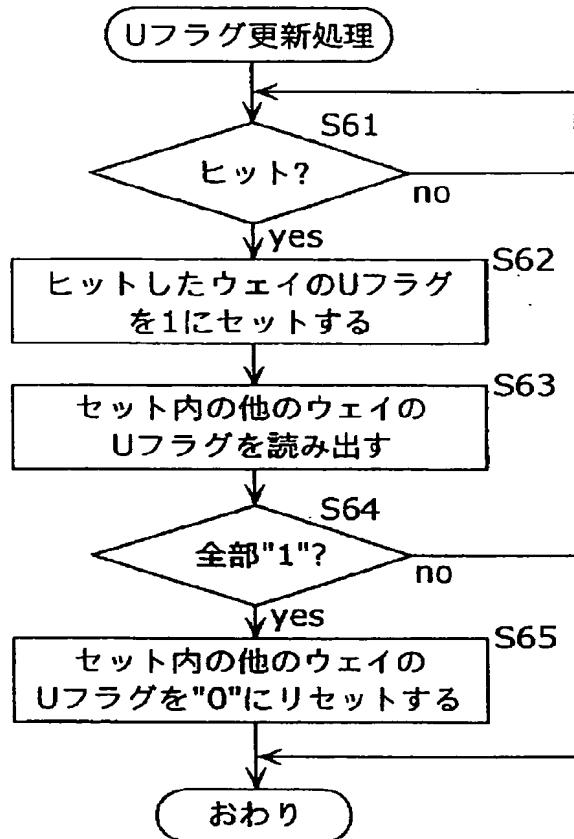
(a)



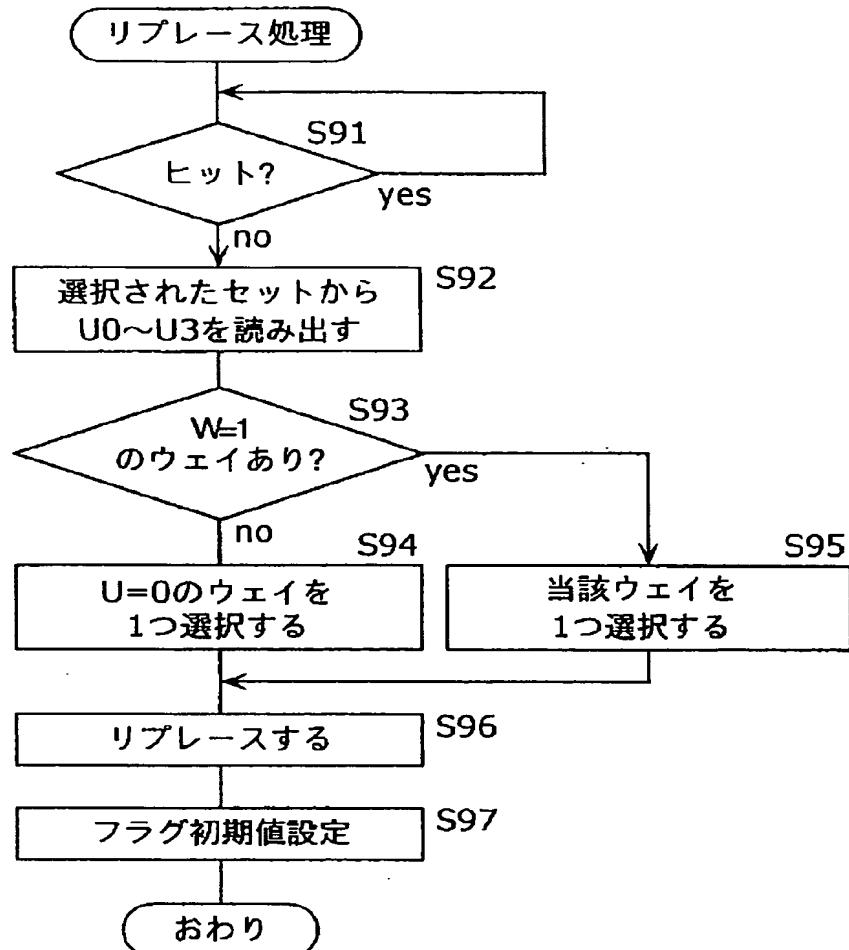
(b)



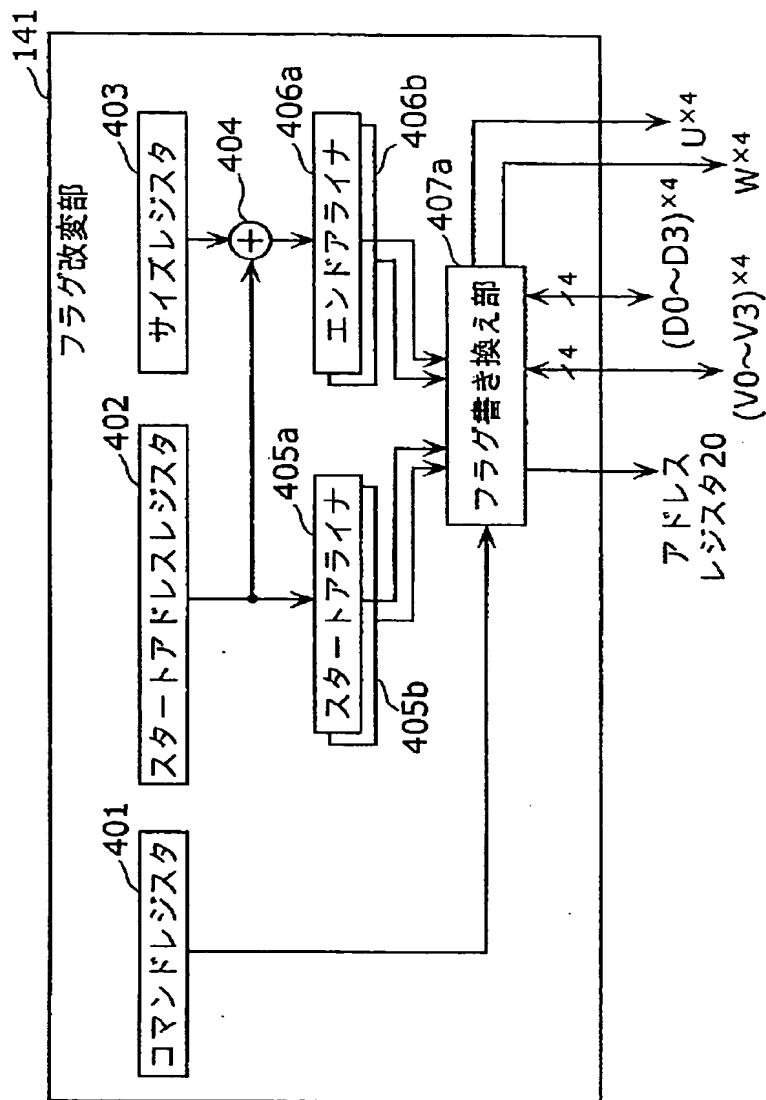
[図14]



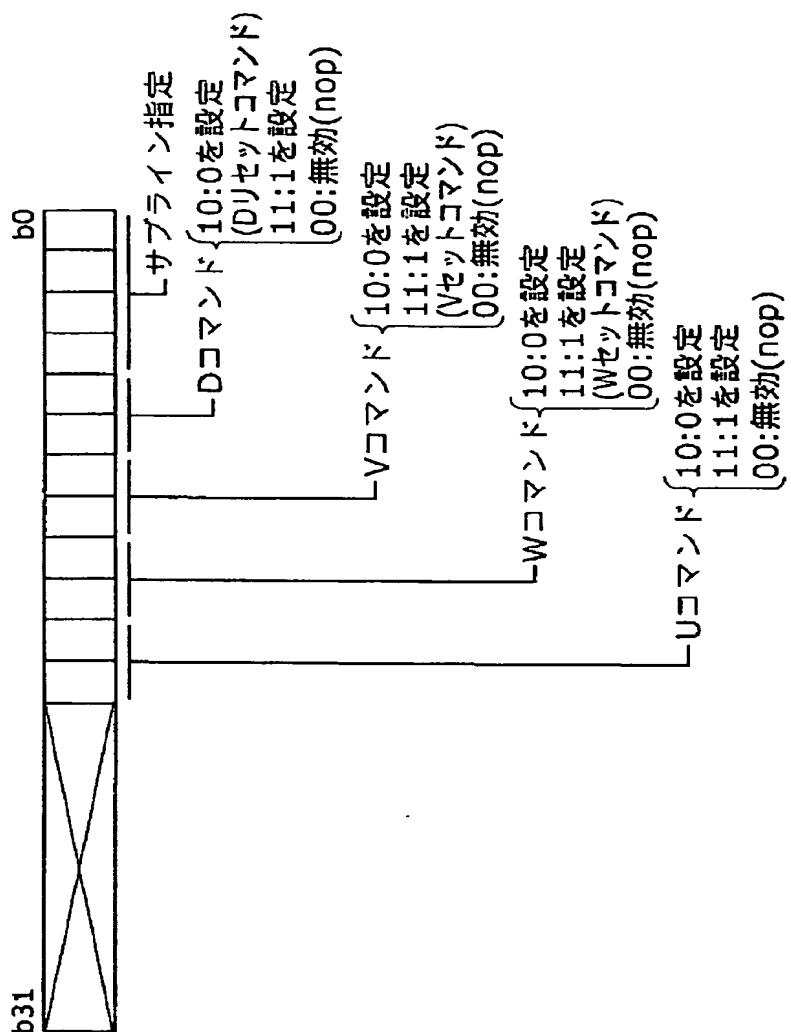
[図15]



[図16]



[図17]



[図18]

